Managing Class Imbalance

BA810: Supervised Machine Learning

Nachiketa Sahoo

Recap

1. Load and explore, training-test split
2. Consider data cleaning and preparation
   1. Imputation, feature selection, …
3. Create pipelines
4. Build and evaluate **predictive models**
   1. Linear models, k-NN, SVM, DTree, …
   2. Ensembles
5. Finetune hyperparameters of the most promising model
   1. Grid/Random search w. Halving
6. Estimate error on unused test-data

Ensembles

- Train multiple models (“weak learners”) and aggregate predictions (mean, mode, etc.)
  - Stabilizes prediction if models are <u>uncorrelated</u>
- Bagging (Bootstrap Aggregating)
  - Learn models from bootstrapped samples
  - Random forest: combine trees, each node considers random subset of columns
- Boosting: learn models sequentially
  - Prioritize records that were poorly predicted
- Voting: mode/mean predicted
  - Stacking: estimated scores/probability to outcome mapping is learnt
    - For some ranges of its score, the strongest model may be weaker than others (alone or combined)

Class Imbalance

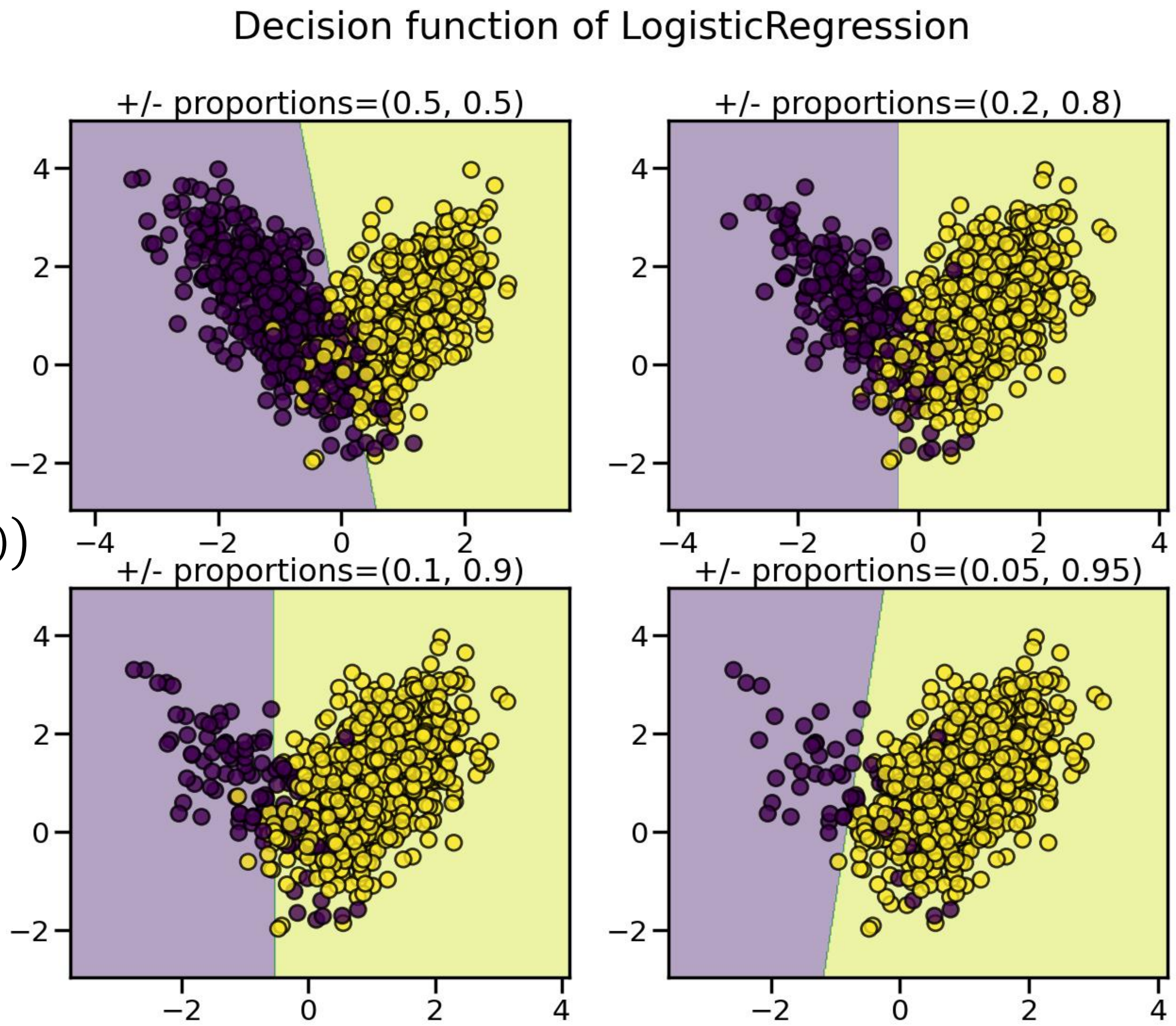
- What is it?
  - Uneven proportion of records from different classes
  - Mild:20–40%, Moderate:1–20%, extreme: <1%
- What's the problem?
  - Objective functions give each record equal weight

$$\min_{\beta} \sum_{i \in N_-} Error\big(y_i, f(x_i;\beta)\big) + \sum_{i \in N_+} Error\big(y_i, f(x_i;\beta)\big)$$

where, N_- and N_+ are the sets of negative and positive training records, respectively

→ biases to predict majority class more often

  - High accuracy but might predict poorly on the minority class (often of more interest, the positive)
    - Frauds, disease positive cases, etc.



Based on example from https://imbalanced-learn.org/

How to Handle?

Use the right metric

- Accuracy gives each *record* equal weight (thus much less weight to minority class)
  - $Accuracy = \frac{\boldsymbol{TN}+TP}{\boldsymbol{TN}+TP+FP+FN}$
  - Balanced accuracy gives each *class* equal weight
    - $\frac{1}{2}\left(\frac{\boldsymbol{TN}}{\boldsymbol{TN}+FP}+\frac{TP}{TP+FN}\right)$
- If you have cost of false negative vs false positive, use that!
  - See our lecture/labs on cost-sensitive evaluations
- These will highlight problems with default objective
  - Low balanced accuracy, high cost due to false negative, …

  (though not fix them)

	PREDICTED CLASS		
ACTUAL CLASS		Positive	Negative
	Positive	TP	FN
	Negative	FP	TN

How to Handle?

Fix the objective

- If we care about records of minority class more, reflect that in the objective function

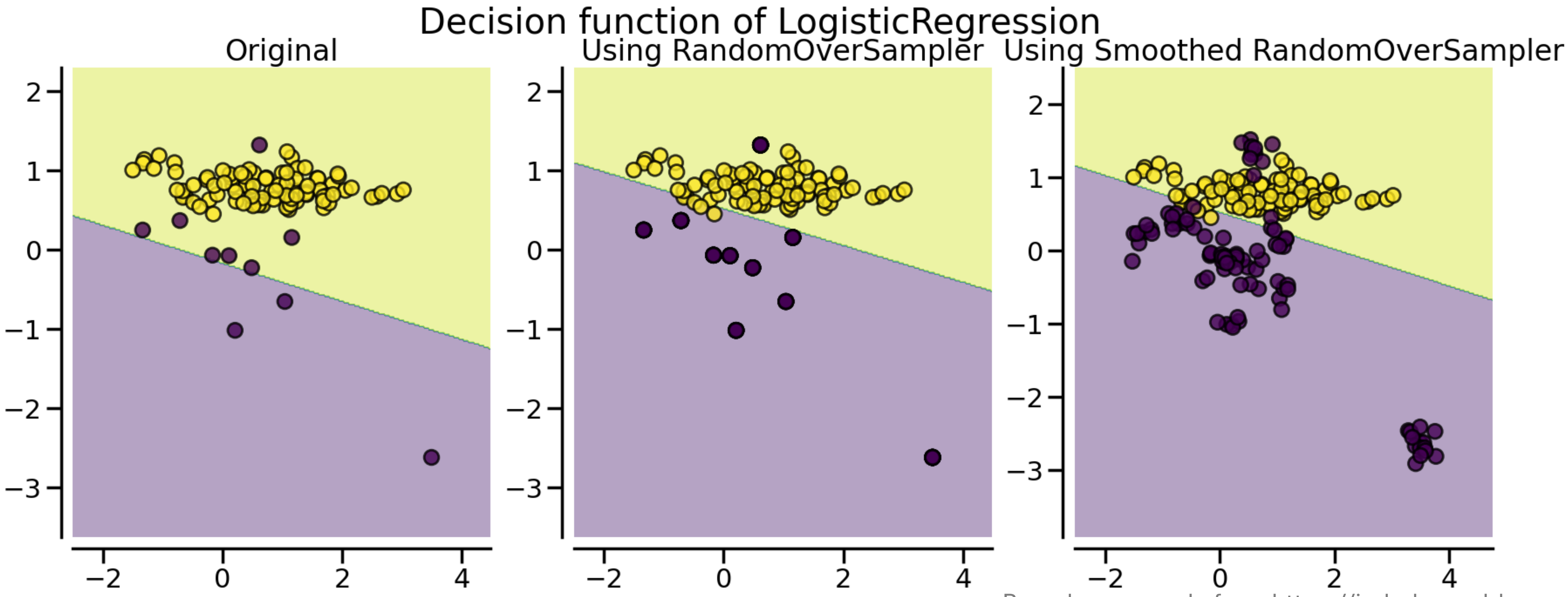
$$\sum_{i \in N_-} Error\big(y_i, f(x_i; \beta)\big) + w \sum_{i \in N_+} Error\big(y_i, f(x_i; \beta)\big)$$

  - Multiply by a class weight w: 5, 10, …
    - Could be the cost of False Negative relative to False Positive
    - Or a value that gives both class equal weight in objective: $\left(\frac{|N_-|}{|N_+|}\right)$
  - Easy to do; many models naturally support this—little added computational cost
- Generally, reduces accuracy, but improves balanced accuracy

How to Handle?

What if the model doesn't support w? *Change Data*

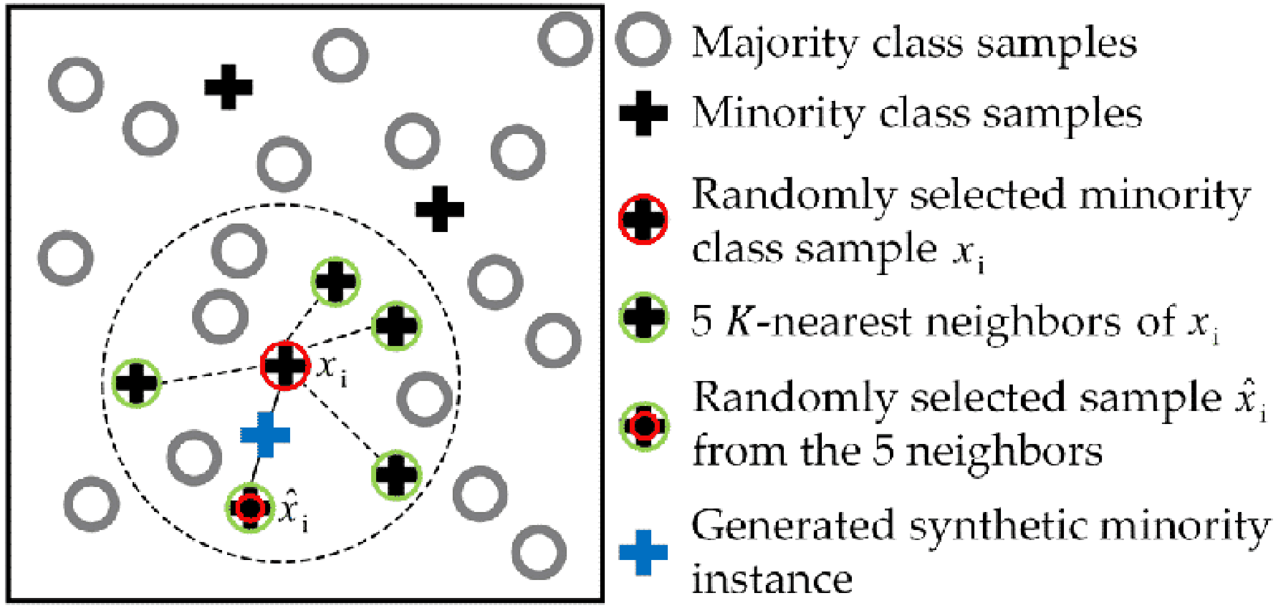
- Oversample minority class: sample with replacement to make class sizes equal
  - Create duplicate minority records or copies with noise added (learn the model from modified data)



Based on example from https://imbalanced-learn.org/

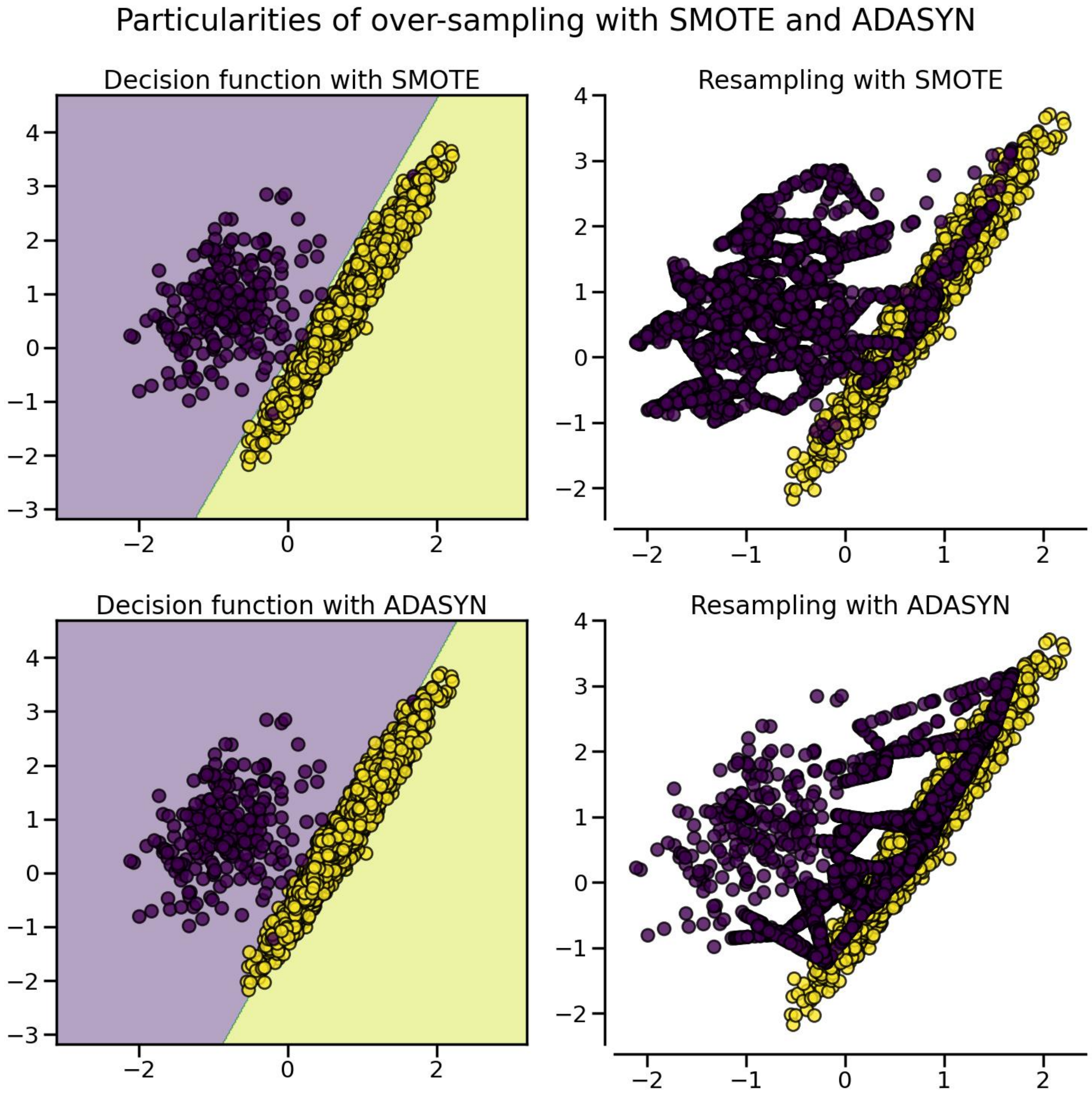
How to Handle?
Change Data

- Synthetic Minority Over-sampling Technique: (SMOTE)



  - Generate more around those points that are at the decision boundaries (ADASYNC and SMOTE variants)

(source: https://rikunert.com/smote_explained)



Based on example from https://imbalanced-learn.org/

How to Handle?
Other approaches

- When dataset is very large … computing time and resource is the constraint
  - Under-sample the majority class
- Learn a standard classifier, but change the decision/probability threshold
  - Pick one that optimizes a desired objective (total cost, net benefit, or balanced accuracy, etc.)
  - Directly changes the location of decision boundary, countering the bias
  - See Lab 4
- Let’s apply these in a default prediction problem with extreme class imbalance.
  - Same one used in Lab 4: 3.33% default and 96.67% don’t.
  - Open Lab 12.

Topics Recap
(With an eye on the final exam)

1. General Machine Learning concepts
   - Supervised/unsupervised, prediction vs causal inference, training/test split
2. Basic predictive models
   - Linear regression (R^2, coefficient, p-value)
   - Classification (models, metrics, cost-based evaluation, …)
   - Bias/variance, underfitting/overfitting
3. Model selection
   - Cross validation, regularization
   - Hyper-parameter search
4. End-to-end Machine Learning process
   - Preprocessing, use of pipelines
5. Specific predictive models
   - Support Vector Machines, Decision Tree, Ensembles
6. Managing imbalanced data in practice
   - Use of appropriate metrics, using class weight, over/under sample to balance

Tips for the Final Exam

- Prioritize lectures and labs, then textbook chapters
  - Don't worry about more complex math, but you should know the conceptual differences between methods and metrics (e.g., decision tree vs logistic regression, cross validation vs bootstrap, etc.)
  - Should know how to compute precision, recall, etc.
- Measurement metrics and interpretations are important
  - E.g., how to interpret logistic regression coefficient, or a confusion matrix
- Be precise in your answers to questions asking to explain (conceptual questions)
  - Stick to specified length limits